This Arduino code is designed for an IoT-based greenhouse automation system that monitors and controls environmental conditions using sensors, relays, and cloud integration. Below is an overview of the key functions and components of the code:

**Libraries and Configuration**

**Included Libraries:**

* **ArduinoIoTCloud:** Enables communication with Arduino IoT Cloud.
* **Arduino\_ConnectionHandler:** Manages Wi-Fi connections.
* **DHT:** Reads temperature and humidity data from a DHT11 sensor.
* **time.h:** Handles time synchronization and timezones.

**Secrets:**

* Wi-Fi credentials (SSID and password).
* Device key and login name for Arduino IoT Cloud authentication.

**Pin Definitions:**

* **DHTPIN:** Pin connected to the DHT11 sensor.
* **MOISTURE\_PIN\_1 & MOISTURE\_PIN\_2:** Analog pins connected to moisture sensors.
* **RELAY\_1 & RELAY\_2:** Pins controlling the water pumps.

**Sensor Calibration:**

* Dry and wet values for moisture sensors to map raw sensor readings to percentages.

**Timing Constants:**

* Defines intervals for manual operation, relay activation, and sensor reading.

**Global Variables**

Variables track sensor readings (moisture, temperature, humidity), relay states, and timing for manual and automatic modes:

* **manual:** A flag to toggle between manual and automatic modes via Arduino Cloud.

**Setup Function**

* Initializes serial communication, pins, sensors, and cloud properties.
* Configures time synchronization with NTP servers.
* Ensures relays are off at startup.

**Loop Function**

* Continuously updates cloud variables and checks the system’s mode:
  + **Manual Mode:** Activates both pumps for a fixed duration (30 seconds).
  + **Auto Mode:** Controls pumps based on moisture sensor readings and thresholds.

**Core Functionalities**

**Manual Mode:**

* Activates both water pumps for 30 seconds.
* Logs the remaining time to the console.

**Auto Mode:**

* Dynamically adjusts thresholds based on "ripeness" values.
* Activates pumps when moisture levels fall below the threshold, with a cooldown period to prevent over-activation.

**Sensor Readings:**

* Averages multiple analog readings to improve accuracy.
* Maps raw sensor data to percentages (0–100%) using calibration values.
* Reads temperature and humidity from the DHT11 sensor.

**Relay Control:**

* Functions to start and stop relays with cooldown logic to avoid frequent relay activation.

**Time and Logging:**

* Logs current time, sensor data, and relay states to the serial monitor.
* Limits logging frequency to prevent spamming.

**Threshold Adjustment:**

* Changes the threshold dynamically based on ripeness levels to optimize irrigation.

**Cloud Integration**

* **Properties for cloud monitoring and control:**
  + **Read-only:** Temperature, humidity, and moisture readings.
  + **Read-write:** Ripeness and manual mode status.
* Synchronizes cloud properties with Arduino IoT Cloud.

**Key Features**

* **Cloud-Enabled Control:** Allows manual mode and ripeness adjustment through Arduino IoT Cloud.
* **Dynamic Thresholds:** Adjusts irrigation based on crop ripeness.
* **Non-Blocking Design:** Ensures sensor readings and relay controls are handled without delaying the main loop.
* **Logging and Debugging:** Logs provide detailed system performance monitoring.

This code framework provides a comprehensive solution for greenhouse automation, offering flexibility for manual overrides, cloud integration, and real-time monitoring.